Computational Visual Media
DOI 10.1007/s41095-xxx-xxxx-x Vol. x, No. x, month year, xx–xx

Article Type (Research/Review)

BLNet: Bidirectional Learning Network for Point Clouds

Wenkai Han[1], **Hai Wu**[1], **Chenglu Wen**[1] ✉ , Cheng Wang[1] and Xin Li[2]

Abstract The key challenge of processing point clouds lies in the inherent unorderness and irregularity of 3D points. By relying on per-point multi-layer perceptions (MLPs), most existing point-based approaches only address the first issue yet ignore the second one. Directly convolving kernels against irregular points will result in loss of shape information. This paper introduces a novel point-based *Bidirectional Learning Network* (BLNet) to analyze irregular 3D points. BLNet optimizes the learning of 3D points through two directions iteratively: feature-guided point shifting and feature learning from shifted points. On the one hand, towards minimum intra-class variances, the points adaptively adjust their positions and converge to a more regular distribution. On the other hand, explicitly modeling point positions leads to a new feature encoding with increased structure-awareness. Then, an attention pooling unit is further designed to selectively combine important features. This bidirectional learning alternately regularizes the point cloud and learns its geometric features, and these two procedures iteratively promote each other towards more effective feature learning. Experiments show that our BLNet is able to learn deep point features robustly and efficiently, and it outperforms prior state-of-the-arts on multiple challenging tasks.

Keywords point clouds, irregularity, shape information, bidirectional learning.

1 Introduction

3D point cloud understanding is critical in many real-world vision applications such as autonomous driving, robotics, and augmented reality. A key challenge for effectively learning point cloud features is that point clouds captured by depth cameras or LiDAR sensors are often unordered and irregular; thus, many effective deep learning architectures [4, 30, 33] are not directly applicable.

To tackle this, many approaches convert irregular point clouds into regular data formats such as multi-view images [9, 25, 32] and 3D voxels [2, 11, 21, 27, 39]. But these conversions result in loss of geometric detail and large memory expense. Alternatively, some recent studies focus on directly processing point clouds. A seminal work, PointNet [24], individually learns per-point features using shared MLPs and gathers a global representation with max-pooling. Although effective, this design ignores local structures that constitute the semantics of the whole object. To solve this problem, many subsequent approaches [15, 20, 26, 37, 42, 45] partition point cloud into nested subsets, then build a hierarchical framework to learn contextual representation from local to global. Nevertheless, these methods perform directly on raw point clouds. The spatial irregularity of point clouds significantly limits their inductive learning performance.

Raw 3D acquisitions typically produce irregular and non-uniformly distributed point clouds. Fig. 1(b) illustrates an example of irregular points sampled from a "square". Suppose we have shared MLPs $\mathcal{G}$ together with their learnable weights $\mathbf{W}$. We apply these convolutions on the points in Fig. 1, then the convolutional output is $\mathbf{f}_x = \mathcal{G}([p_1, p_2, p_3, p_4]_x, \mathbf{W})$, where $x = (a, b)$. The shared point-wise MLPs utilized for encoding points can ensure permutation-invariance and address the unorderness issue. However, due to the irregular sampling in (b), we usually get $\mathbf{f}_a \neq \mathbf{f}_b$. Therefore, local features extracted on noisy or

1 School of Informatics, Xiamen University, 422 Siming South Road, Xiamen 361005, China. E-mail: hlxwk0525@gmail.com, wuhai@stu.xmu.edu.cn, clwen@xmu.edu.cn, cwang@xmu.edu.cn.

2 School of Electrical Engineering and Computer Science, Louisiana State University, Baton Rouge, LA 70803, USA. E-mail: xinli@cct.lsu.edu.

Manuscript received: 2014-12-31; accepted: 2015-01-30.

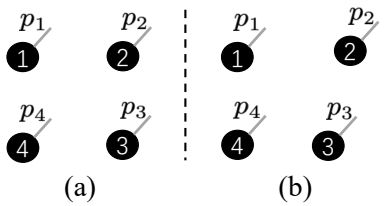

Fig. 1 (a) is ideally regularly sampled points and (b) is irregularly sampled points. Each point in (a, b) is associated with coordinates p.

irregularly sampled points are often unstable, causing loss of shape information. Our observation is that: (1) If the sampling of (b) becomes more regular, its learned feature will converge to $\mathbf{f}_a$ and become more stable. (2) On the other hand, if a more accurate feature description (e.g., $\mathbf{f}_a$) is available, to deform the shape (e.g., shape (b)) according to this feature will allow the point cloud to adjust their points towards a more regular distribution. Thus, making sampled points more regular and obtaining more accurate features arise as important issues to address.

To this end, we formulate BLNet, the first work to apply *bidirectional learning* to point clouds and analyze irregular 3D points through bidirectional interaction between points and features. The key to BLNet is to capitalize on two directions iteratively: feature-guided point shifting and feature learning from shifted points. On the one hand, taking the task loss as *feedback*, the Position Feedback module associated with adaptive 3D displacements is proposed to automatically adjust the positions of points. Via minimizing intra-class variances, 3D points are regularized towards a certain distribution that fits the network well. On the other hand, we present a new Feature Modeling module, which explicitly encodes point positions with increased structure-awareness; and, we further design an attention pooling to selectively focalize and combine important features. This bidirectional learning alternately regularizes the point cloud and learns its geometric features, and these two procedures iteratively promote each other towards more effective feature learning. Extensive experiments verified the superiority of our BLNet on multiple challenging datasets including ModelNet40 [39], ShapeNet Parts [44], S3DIS [1] and ScanNet [2]. Moreover, we show ablation experiments and visualization for a better understanding of BLNet.

2 Related work

Deep Learning on Regular Domain. To leverage the impressive success of traditional convolution in the regular formats (*i.e.*, images), extensive approaches usually transform irregular point clouds into regular data formats such as multi-view images and 3D voxels. For the former, view-based methods [9, 25, 32] render multiple images from point clouds based on different views and apply standard CNNs on rendered images. For the latter, voxel-based methods [2, 11, 21, 27, 39] structure point cloud using 3D regular voxels. Afterwards, 3D CNNs can be directly applied similarly to images. However, both regular formats need projective or voxelized transformations, which yield a quantifiable loss of geometric information. By contrast, our BLNet focuses on directly processing point clouds and does not rely on additional transformations.

Deep Learning on Irregular Domain. Inspired by the pioneering work PointNets [24, 26], many recent methods directly process point clouds and design sophisticated networks to capture features. These approaches can be generally classified as 1) neighbouring feature pooling [8, 26, 42, 45, 46], 2) graph message passing [10, 36, 37], 3) kernel-based convolution [15, 19, 29, 35, 38]. These methods directly run on raw point clouds. The spatial irregularity of point clouds limits their inductive learning and further promotion.

Bidirectional Learning. It has been shown effective for enhancing the performance of uni-directional learning in multiple tasks such as language translation [22, 40], image generation [23], and image translation [28]. It utilizes additional top-to-down (target-to-source) training to reduce the uni-directional dependency between source and target. However, they typically train the two directions separately and fuse them at the end. In contrast, our BLNet is the first work to achieve bidirectional interaction between points and features on point clouds. And it alternately combines two learning directions, consequently forming a complete network for training.

3 Our approach/method

Locality. Most recent point cloud learning frameworks [15, 26, 42] are trained to extract representations based on local features, which has been shown more effective than earlier work [24] that learns global description. Similarly, we aim to design our network and modules locally.

Bidirectional Learning. The position feedback module performs feature-guided point shifting towards a more regular distribution, and the feature modeling module explores discriminative features from shifted points. To realize our bidirectional learning pipeline, we integrate a position feedback module and a feature

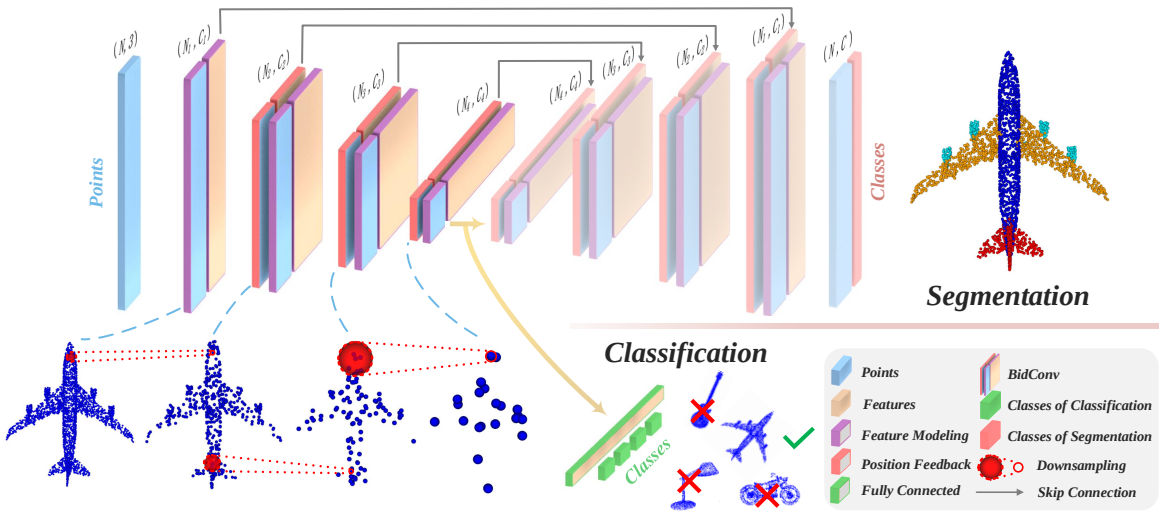

Fig. 2 BLNet architecture. For classification, we use four BidConv before the fully connected classifier. For segmentation, we follow a U-net architecture. Points downsampling (using FPS in [26]) and upsampling are also included in our convolution, depending on its use. $N_1 > N_2 > N_3 > N_4$ denotes points downsampling in each convolution, and $C_1 < C_2 < C_3 < C_4$ denotes dimension-increasing feature channels at each point. Note that the first BidConv does not include position feedback module, as there are no features extracted from points at the beginning of the network.

modeling module into one bidirectional convolution operator, namely, BidConv. Stacking multiple BidConv enables the two modules to execute alternately and promote each other, as illustrated in Fig. 2.

BLNet Architecture. As shown in Fig. 2, we build a hierarchical framework, BLNet, which can be applied to multiple tasks including point cloud classification and segmentation. In both tasks, we use four BidConv to learn dimension-increasing features with progressive downsampling. The final global representation followed by fully connected layers is configured for classification tasks. For segmentation tasks, high-resolution point-wise predictions are required, and this can be realized by designing deconvolution; we still utilize BidConv to recover resolution, and progressively upsample the compact features obtained from encoder until the original resolution. Higher-resolution points are forwarded from earlier corresponding convolution layers, following the coarse-to-fine design of U-Net [47]. Inspired by [17], features at the same resolution are skip-connected to preserve previous information. Both classification and segmentation models can be trained in an end-to-end manner.

3.1 Feature Modeling

In order to learn more accurate features, and use them to guide the shifting of points, we develop a new feature modeling module. This module includes a position encoding unit and an attention pooling unit, which can more discriminatively capture and combine features.

Position Encoding. Given a point cloud P together with corresponding point features (e.g., raw RGB, or intermediate learned features), this unit aims to explicitly encode the spatial layout of 3D points, which plays a crucial role in shape analysis. Existing approaches [26, 37, 45] typically concatenate position information with point features, then transform the concatenated results for feature learning. But these approaches are suboptimal on capturing meaningful geometry patches. In contrast, we perform an explicit encoding of point positions first and then combine the output with point features for further enhancement. This enables each 3D point to observe its local geometry, thus eventually enriching the entire network with increased structure-awareness. Particularly, this unit consists of the following steps:

Locality. For the i-th point, to increase its receptive fields, we index its neighboring points with dilated K-Nearest Neighbours (KNN) algorithm. Specifically, we sample K points at equal intervals from the top $K \times r$ neighboring points, where r denotes the dilation rate.

Position Encoding. For each of the K neighboring points $\{p_i^1 \cdots p_i^k \cdots p_i^K\}$ of the center point p_i, we explicitly encode the relative position as follows:

$$\mathbf{r}_i^k = \mathcal{G}_r(p_i^k - p_i, \mathbf{W}_r), \tag{1}$$

where p_i and p_i^k are global x-y-z coordinates of points, and $p_i^k - p_i$ denotes relative coordinates, which can retain translation invariance. $\mathcal{G}_r()$, together with its learnable weights $\mathbf{W}_r$, denotes a shared function. This

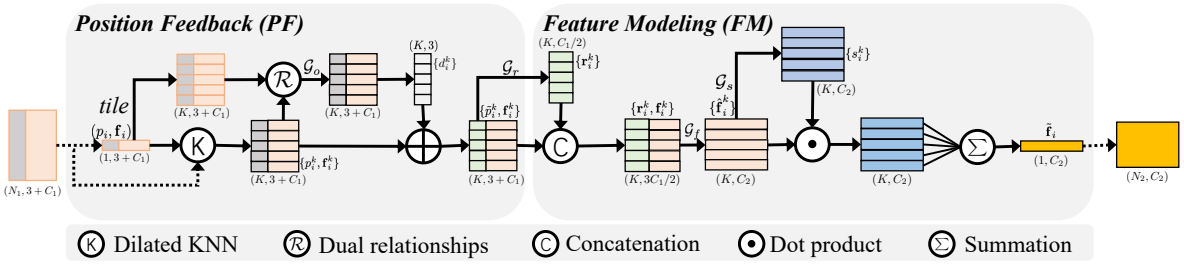

Fig. 3 BidConv with downsampling ($N_2 < N_1$).

function can be implemented with any differentiable architecture, and we use point-wise shared MLPs in this work to address the unorderness issue of 3D points. Note that all the functions in our paper are complemented by per-point MLPs, and we omit explaining this in the remaining of this paper.

Feature Enhancement. The prior semantic information contained in per-point features can further enhance the distinctiveness of learned position features. For each neighboring point p_i^k, we concatenate its position features $\mathbf{r}_i^k$ with corresponding point features $\mathbf{f}_i^k$, then use a shared function $\mathcal{G}_f()$ to combine them. Thus, we obtain the enhanced feature vector $\hat{\mathbf{f}}_i^k$ with the following formulation:

$$\hat{\mathbf{f}}_i^k = \mathcal{G}_f([\mathbf{r}_i^k, \mathbf{f}_i^k], \mathbf{W}_f), \tag{2}$$

where $[,]$ is the concatenation operation.

Attention Pooling. This unit is used to aggregate the set of neighboring point features. Different features in the local region impose varied impacts on the local representation. The leading strategy in existing literatures [26, 37] for integrating the neighboring features is by employing max/mean pooling. Yet this frequently results in a loss of useful information. In contrast, we design a new attention pooling to selectively focus on the most relevant features. In particular, this unit includes the following steps:

Attention Scores. Given the set of local features $\hat{\mathbf{F}}_i = \{\hat{\mathbf{f}}_i^1 \cdots \hat{\mathbf{f}}_i^k \cdots \hat{\mathbf{f}}_i^K\}$, we design a shared function $\mathcal{G}_s()$ to learn a unique attention score for each channel of point features. Differently, to make weight coefficients comparable among different channels, this function consists of shared MLPs followed by a channel-level $softmax$. It is formulated as:

$$s_i^k = \mathcal{G}_s(\mathcal{R}_1(\hat{\mathbf{f}}_i, \hat{\mathbf{f}}_i^k), \mathbf{W_s}), \tag{3}$$

where a pairwise function $\mathcal{R}_1$ indicates a high-level relationship between the centroid point and its certain neighbor. Here we define $\mathcal{R}_1$ as: $\mathcal{R}_1(\hat{\mathbf{f}}_i, \hat{\mathbf{f}}_i^k) = |\hat{\mathbf{f}}_i - \hat{\mathbf{f}}_i^k|$, which measures the feature difference between point pairs and guides to assign more attention scores to similar neighbors.

Weighted Summation. We consider the learned attention scores as a soft mask that selectively focalizes important features. Then, the local representation $\tilde{\mathbf{f}}_i$ is obtained by summing weighted features as:

$$\tilde{\mathbf{f}}_i = \sum_{k=1}^{K} \{\hat{\mathbf{f}}_i^k \cdot s_i^k\}. \tag{4}$$

3.2 Position Feedback

Given extracted features, this module aims to perform a feature-guided point shifting. To achieve this, we regress an adaptive 3D displacement for each point by considering its feature, as well as position. Taking the task loss (*i.e.*, cross-entropy loss) as *feedback*, these displacement can learn to adjust the positions of points. Via minimizing intra-class variances (i.e., minimizing feature difference of intra-class points), local points are shifted towards a certain distribution that fits the network well. With respect to original irregular points, this distribution is more regular and leads to more effective feature learning. Concretely, this module comprises the following steps:

3D Displacements. Using dilated KNN, we index K neighboring points of the center point p_i as $\{p_i^1 \cdots p_i^k \cdots p_i^K\}$, together with corresponding point features $\{\tilde{\mathbf{f}}_i^1 \cdots \tilde{\mathbf{f}}_i^k \cdots \tilde{\mathbf{f}}_i^K\}$. For each of them, we define a dual relation of spatial and semantic levels to generate point-wise 3D displacement. It is formulated as:

$$d_i^k = \mathcal{G}_o([\mathcal{R}_{spa}(p_i, p_i^k), \mathcal{R}_{sem}(\tilde{\mathbf{f}}_i, \tilde{\mathbf{f}}_i^k)], \mathbf{W}_o), \tag{5}$$

where $\mathcal{R}_{spa}(p_i, p_i^k) = (p_i - p_i^k)$ indicates a spatial relation between point p_i^k and its center, and $\mathcal{R}_{sem}(\tilde{\mathbf{f}}_i, \tilde{\mathbf{f}}_i^k) = |\tilde{\mathbf{f}}_i - \tilde{\mathbf{f}}_i^k|$ is defined in a formulation similar to $\mathcal{R}_1$. The first term $\mathcal{R}_{spa}$ enables the corresponding point features are always aware of their relative spatial locations, and the second term $\mathcal{R}_{sem}$ helps to learn shift distance according to feature difference. For each 3D displacement, the former provides direction and the latter obtains magnitude, which jointly guide the irregular neighbors to meaningful patches under the *feedback* of the task loss.

Position Updating. The learned adaptive

displacements can be viewed as semantics-driven factors. With them, the points are automatically shifted, generally towards a more regular distribution. Formally, each neighboring point can be updated as follows:

$$\tilde{p}_i^k = p_i^k + \alpha \cdot \mathcal{N}(d_i^k), \tag{6}$$

where α is a scale coefficient, $\mathcal{N}(\cdot)$ is a normalization function mapping the value to range $[-1, 1]$, and $\tilde{p}_i^k$ is shifted global coordinates. Considering that the network is very fragile at the beginning of training, α is initialized as 0 and gradually assigned an appropriate weight to adapt to local structures.

As shown in Fig. 2, a position feedback module utilizes features extracted from the previous feature modeling module to perform feature-guided point shifting. With iterative downsampling and upsampling, these two modules need to run on different resolutions. We integrate the two consecutive modules operated on the same resolution together as one operation, BidConv, as shown in Fig. 3. And we utilize KNN only once after each sampling. Stacking multiple BidConv enables the two modules to execute alternately and promote each other, thereby forming an effective learning network.

4 Results and discussion

Datasets. We evaluated on four datasets for multiple tasks ranging from shape classification (ModelNet40 [39]), part segmentation (ShapeNet Parts [44]), to semantic segmentation (S3DIS [1] and ScanNet [2]). The experiment setting for each dataset is listed below:

- ModelNet40: 12,311 3D mesh models of 40 object categories. We followed an official split with 9,843/2,468 models for training and testing.
- ShapeNet Parts: 16,881 CAD models from 16 object categories. Each point is annotated with a certain one of 50 part classes and each point cloud consists of 2 to 5 parts. Officially, we split this dataset with 14,006 objects for training and 2,874 for testing.
- ScanNet: 1,512 reconstructed indoor scenes with 21 semantic categories. We split this dataset with 1,201/312 scenes for training and testing.
- S3DIS: 271 real rooms from three different buildings with 13 semantic categories. Following the experiment setttings in [24, 34], we used two dominant settings for training and testing, *i.e.*, *6-fold* and *Area-5* cross validation.

Tab. 1 Quantitative results (%) of different approaches on ModelNet40.

Methods	Input	#Points	OA
PointNet [24]	xyz	$1k$	89.2
PointNet++ [26]	xyz	$1k$	90.7
KCNet [29]	xyz	$1k$	91.0
RSCNN [19]	xyz	$1k$	91.7
DGCNN [37]	xyz	$1k$	92.2
PointCNN [15]	xyz	$1k$	92.2
Point2Sequence [18]	xyz	$1k$	92.6
A-CNN [12]	xyz	$1k$	92.6
PointASNL [43]	xyz	$1k$	92.9
Grid-GCN [41]	xyz	$1k$	93.1
ShellNet [46]	xyz	$1k$	93.1
InterpCNN [7]	xyz	$1k$	93.0
Point2Node [3]	xyz	$1k$	93.0
BLNet (Ours)	**xyz**	**1k**	**93.5**
KPConv [35]	xyz	$6k$	92.9
PointWeb [45]	$xyz, normal$	$1k$	92.3
PointConv [38]	$xyz, normal$	$1k$	92.5
BLNet (Ours)	**xyz, normal**	**1k**	**93.7**
PointNet++ [26]	$xyz, normal$	$5k$	91.9
SO-Net [14]	$xyz, normal$	$5k$	93.4

4.1 Shape Classification

We evaluated our network on classifying point clouds sampled from ModelNet40 [39]. Using a widely-used sampling density in existing literatures, we uniformly sampled 1,024 points from each 3D mesh model and normalized them to a unit sphere. We used Overall Accuracy (OA) as the evaluation metric. Also, to reduce over-fitting, we employed the dropout technique [31] with 80% ratio in the penultimate FC layers.

Results. For fair comparisons, we presented overall shape accuracy as well as input settings in Table 1. BLNet clearly surpasses all previous approaches in terms of different input settings. Specifically, when only using xyz information, BLNet achieves an OA of 93.5% and outperforms a set of representative methods such as PointNet++ [26] (90.7%), PointCNN [15] (92.2%), RSCNN[1] [19] (91.7%), and Point2Node [3] (93.0%), especially KPConv [35] (92.9%) using more ($6k$) points. Moreover, when additionally using normal information, BLNet achieves a higher accuracy of 93.7% and significantly outperforms all existing methods, particularly including PointNet++ [26] (91.9 %) and SO-Net [14] (93.4%), which use more ($5k$) points.

[1]Only the single scale RSCNN [19] is released so far.

Tab. 2 Quantitative results (%) of different approaches on S3DIS, ShapeNet Parts, and ScanNet.

Methods	S3DIS *6-fold*		S3DIS *Area-5*		ShapeNet Parts	ScanNet
	OA	mIoU	OA	mIoU	mcIoU	OA
PointNet [24]	78.6	47.6	-	41.1	80.4	73.9
PointNet++ [26]	81.0	54.5	-	-	81.9	84.5
KCNet [29]	-	-	-	-	82.2	-
DGCNN [37]	84.1	56.1	-	-	82.3	-
RSCNN [19]	-	-	-	-	84.0	-
RSNet [6]	-	56.5	-	51.9	81.4	-
PointCNN [15]	88.1	65.4	85.9	57.3	84.6	85.1
SPGraph [13]	85.5	62.1	86.4	58.0	-	-
A-CNN [12]	87.3	62.9	-	-	83.0	85.4
PointWeb [45]	87.3	66.7	86.9	60.3	-	85.9
ShellNet [46]	87.1	66.8	-	-	82.8	85.2
Point-Edge [16]	88.2	67.8	87.2	61.8	-	-
ELGS [42]	87.6	66.3	88.4	60.1	-	85.3
Grid-GCN [41]	-	-	86.9	57.8	-	85.4
PointASNL [43]	88.8	68.7	87.7	62.6	83.4	-
RandLA-Net [5]	87.2	68.5	-	-	-	-
BLNet (Ours)	**89.3**	**70.8**	**89.1**	**64.2**	**85.1**	**86.7**

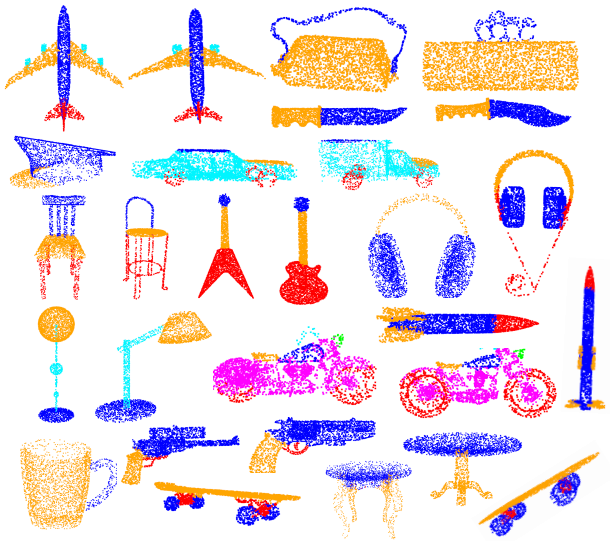

Fig. 4 Part segmentation results from ShapeNet Parts.

4.2 Part Segmentation

Part segmentation is a challenging task for fine-grained shape analysis. We evaluated our network on ShapeNet Parts [44] benchmark and randomly sampled 2,048 points as the input. We reported class average IoU (mcIoU) as the evaluation metric.

Results. As in Table 2, our BLNet achieves the best performance with mcIoU of 85.1%. This considerably outperforms other competitive baselines, *i.e.*, DGCNN [37] (82.3%), RSCNN [19] (84.0%), and the recent PointASNL [43] (83.4%). Fig. 4 shows some qualitative visualization examples of different shapes. BLNet can segment distinctive parts from diverse shapes.

4.3 Semantic Segmentation

This task takes 3D point clouds as input and assigns one semantic class label for each point. We evaluated our network on two datasets: S3DIS [1] and ScanNet [2]. Following PointCNN [15], we first split points according to the room and sliced the rooms into 1.5m by 1.5m blocks, with 0.3m padding on each side. We sampled 2,048 points for each block in training, and adopted all the points for evaluation during testing. For S3DIS, we adopted two widely-used evaluation metrics: Overall Accuracy (OA) and mean class IoU (mIoU). For ScanNet, to make a fair comparison with other approaches, we did not use RGB information and converted the segmentation results from the testing data into semantic voxel labeling for evaluation. Here we reported voxel-wise Overall Accuracy (OA) as the evaluation metric.

Results on S3DIS. As in Table 2, our BLNet outperforms all previous approaches in terms of two evaluation settings. Specifically, as for *6-fold* mIoU, BLNet significantly surpasses prior start-of-the-arts, *i.e.*, PointNet++ [26] (16.3↑), PointCNN [15] (5.4↑), ELGS [42] (4.5↑), and the recent RandLA-Net [5] (2.3↑); as for *Area-5* mIoU, BLNet considerably outperforms PointNet [24] (23.1↑), PointCNN [15] (6.9↑), and also outperforms the recent Grid-GCN [41] (6.4↑), PointASNL [43] (1.6↑). In addition, we listed detailed mIoU results of all the categories in Table 3. BLNet achieves better or on par performance with respect to other competitive approaches [5, 13, 15, 24, 42, 45]. It is worth mentioning that BLNet works well in some similar geometric shapes (e.g., column *vs.* wall) and the categories with various shapes (e.g., chair and table). This indicates that BLNet captures discriminative shape representations from complex structures. Some qualitative visualization results on different real scenes are given in Fig 5.

Results on ScanNet. As in Table 2, when doing semantic voxel labeling task, BLNet achieves better performance (86.7%) than existing methods, *i.e.*, PointNet++ [26] (84.5%), PointCNN [15] (85.1%), ELGS [42] (85.3%), and the recent Grid-GCN [41] (85.4%).

4.4 Ablation Study

To validate the contribution of each module in our framework, we conduct ablation studies in Table 4. These experiments are evaluated on S3DIS [1] with *Area-5* cross validation and reported mIoU as the standard metric.

(1) Remove position encoding. Position encoding enables each 3D point to observe its local geometry. After removing this unit, we directly transform point features using per-point MLPs and

Tab. 3 All categories results (%) of different approaches on S3DIS *6-fold*.

Methods	mIoU	ceil.	floor	wall	beam	col.	wind.	door	table	chair	sofa	book.	board	clut.
PointNet [24]	47.6	88.0	88.7	69.3	42.4	23.1	47.5	51.6	54.1	42.0	9.6	38.2	29.4	35.2
SPGraph [13]	62.1	89.9	95.1	76.4	62.8	47.1	55.3	68.4	69.2	73.5	45.9	63.2	8.7	52.9
PointCNN [15]	65.4	**94.8**	**97.3**	75.8	63.3	**51.7**	58.4	57.2	69.1	71.6	**61.2**	39.1	52.2	58.6
ELGS [42]	66.3	93.7	95.6	76.9	42.6	46.7	63.9	**69.0**	70.1	76.0	52.8	57.2	54.8	62.5
PointWeb [45]	66.7	93.5	94.2	80.8	52.4	41.3	64.9	68.1	71.4	67.1	50.3	62.7	62.2	58.5
RandLA-Net [5]	68.5	92.7	95.6	79.2	61.7	47.0	63.1	67.7	68.9	74.2	55.3	63.4	63.0	58.7
BLNet (Ours)	**70.8**	94.3	97.1	**83.5**	**64.1**	50.5	**72.5**	63.2	**74.2**	**79.5**	50.7	**64.0**	**63.6**	**62.7**

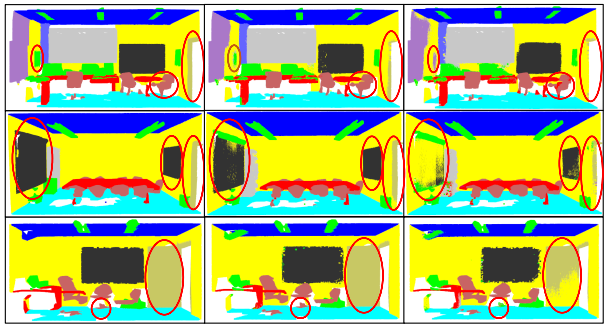

Fig. 5 Qualitative results from S3DIS. From left to right are ground truth, BLNet (Ours), and PointCNN [15], respectively. The segmentation results of BLNet is closer to the ground truth than that of PointCNN.

feed the output features into the subsequent attention pooling. As shown in Table 4, removing position encoding causes a significant performance drop. This is because the spatial distributions of points play a crucial role in 3D shape analysis, and our position encoding unit can effectively increase structure-awareness via explicitly encoding relative point positions.

Tab. 4 The mIoU results (%) of all ablated networks based on our full BLNet.

	mIoU
(1) Remove position encoding	54.90
(2) Replace with mean-pooling	58.72
(3) Replace with max-pooling	60.55
(4) Remove position feedback	59.70
(5) The Full framework (BLNet)	**64.17**

(2˜3) Replace attention pooling with max/mean pooling. Attention pooling learns to selectively focalize important features and then combine them. To make comparisons, we replace this unit with widely-used max/mean pooling. As Table 4 shows, our attention pooling considerably surpasses max/mean pooling. This demonstrates that attention pooling is able to keep important features and gather a discriminative representation.

(4) Remove position feedback. This module aims to adaptively shift 3D points towards a more regular distribution. After removing this module, we directly feed original irregular points into the subsequent feature modeling module. As shown in Table 4, removing position feedback considerably harm performance. This indicates that shifted distributions can fit the network better than original ones and promote feature learning. To further verify the effectiveness of position feedback, we visualize T-SNE results of shape features from different categories without and with this module in Fig. 6. Note that since differences in distributions of input points will result in different features (see Sec. 1 Introduction), we visualize features to better view the effect of different input distributions w/o and with position feedback. On the left of Fig. 6, features extracted from irregular points (w/o position feedback) are mixed and less distinguishable from each other, showing that directly consuming raw point clouds could cause more shape information loss. In contrast, on the right of Fig. 6, features extracted from shifted points (with position feedback) can be more easily partitioned. This proves that shifted distributions induced by adaptive 3D displacements can lead to more intra-class consistency and inter-class distinctiveness with respect to original irregular distributions.

Robustness under Noises. We further demonstrate the robustness of our BLNet with respect to two representative baselines, *i.e.*, PointNet++ [26] and PointCNN [15]. When applying random noises (increasing irregularity) from the range $[-0.01, 0.01]$ to each point, the mIoU of BLNet, PointCNN, and PointNet++ on S3DIS *Area-5* [1] decreases by 1.2%, 3.4% and 4.5%, respectively. We find that BLNet is more robust under random noises. This is because the position feedback module generates adaptive 3D displacements to shift each point, which can be effective in resisting random noises.

Model Complexity. Fig. 7 qualitatively shows the complexity comparison of different approaches by Accuracy versus Parameters, and Accuracy versus Latency. Compared with modern competitive methods

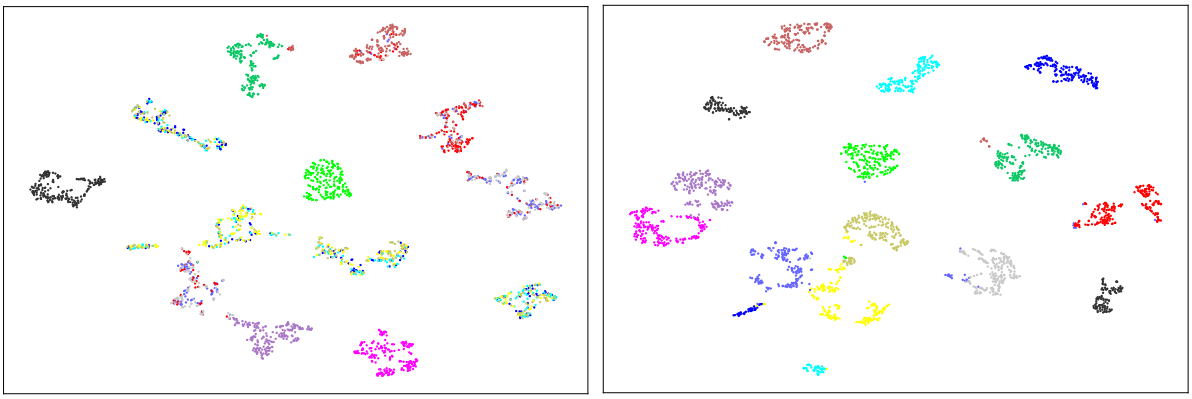

Fig. 6 T-SNE visualization of features w/o (Left) and with position feedback module (Right). Left: red/grey/blue are all mixed up; yellow and cyan are mixed. Right: they are clearly separated. Different (13) colors denote different categories, and these experiments are performed on S3DIS [1].

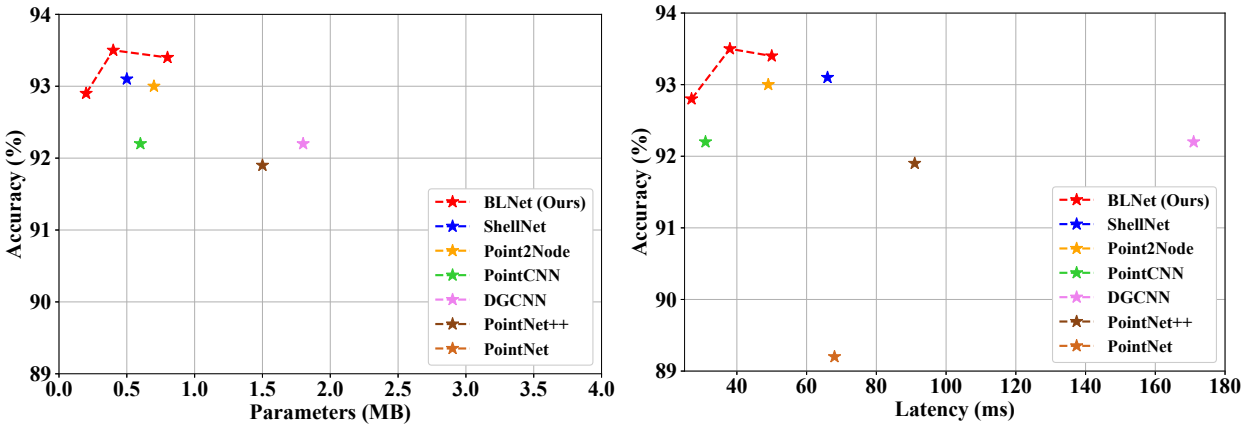

Fig. 7 Comparisons between BLNet and other representative baselines on ModelNet40 [39]. Left: Accuracy *vs.* Parameters. Right: Accuracy *vs.* Latency. All baselines are evaluated on ModelNet40 with batch size 16 and 1024 input points for fair comparisons.

[3, 15, 24, 26, 37, 46], BLNet achieves a significantly better accuracy *vs.* complexity trade-off, thus demonstrating its effectiveness and efficiency. Note that the BLNet model with the highest accuracy (middle) is our original version, and we increase or decrease its model complexity by simply scaling feature channels in Fig. 2.

5 Conclusions

This is the first work to apply *bidirectional learning* to point clouds and achieve bidirectional interaction between points and features. Specifically, we propose a novel point-based Bidirectional Learning Network (BLNet) to analyze irregular 3D points. BLNet optimizes the learning of 3D points through two directions iteratively: feature-guided point shifting and feature learning from shifted points. (1) The position feedback module utilizes adaptive 3D displacements to automatically shift points, leading to a more regular distribution. (2) A new feature modeling module explicitly encodes point positions with increased structure-awareness, and a powerful attention pooling in this module selectively combine important features. These two modules alternately regularize the point cloud and learn its geometric features, and iteratively promote each other towards more effective feature learning. Extensive experiments verified the superiority of our BLNet on various challenging benchmarks.

Acknowledgements

References

[1] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese. 3d semantic parsing of

large-Scale indoor spaces. In *CVPR*, pages 1534–1543, June 2016.

[2] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Niessner. ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 5828–5839, July 2017.

[3] W. Han, C. Wen, C. Wang, X. Li, and L. Qing. Point2Node: Correlation Learning of Dynamic-Node for Point Cloud Feature Modeling. In *AAAI*, 2020.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, June 2016.

[5] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *CVPR*, 2020.

[6] Q. Huang, W. Wang, and U. Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *CVPR*, June 2018.

[7] H. L. Jiageng Mao, Xiaogang Wang. Interpolated convolutional networks for 3d point cloud understanding. *ICCV*, 2019.

[8] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. In *CVPR*, 2019.

[9] E. Kalogerakis, M. Averkiou, S. Maji, and S. Chaudhuri. 3d shape segmentation with projective convolutional networks. In *CVPR*, pages 6630–6639, jul 2017.

[10] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv:1609.02907*, 2016.

[11] R. Klokov and V. Lempitsky. Escape from Cells: Deep kd-networks for the recognition of 3d point cloud models. In *ICCV*, pages 863–872, oct 2017.

[12] A. Komarichev, Z. Zhong, and J. Hua. A-CNN: Annularly convolutional neural networks on point clouds. In *CVPR*, 2019.

[13] L. Landrieu and M. Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *CVPR*, pages 4558–4567, jun 2018.

[14] J. Li, B. M. Chen, and G. H. Lee. SO-Net: Self-organizing network for point cloud analysis. In *CVPR*, page 9397–9406, jun 2018.

[15] Y. Li, B. Rui, M. Sun, and B. Chen. PointCNN: Convolution on x-transformed points. In *NeurIPS*, 2018.

[16] S. L.-X. S. C.-W. F. J. J. Li Jiang, Hengshuang Zhao. Hierarchical point-edge interaction network for point cloud semantic segmentation. *ICCV*, 2019.

[17] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, July 2017.

[18] X. Liu, Z. Han, Y. S. Liu, and M. Zwicker. Point2Sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. In *AAAI*, 2019.

[19] Y. Liu, B. Fan, S. Xiang, and C. Pan. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*, pages 8895–8904, 2019.

[20] Z. Liu, H. Tang, Y. Lin, and S. Han. Point-voxel cnn for efficient 3d deep learning. In *NeurIPS*, 2019.

[21] D. Maturana and S. Scherer. VoxNet: A 3d convolutional neural network for real-time object recognition. In *IROS*, pages 922–928, sep 2015.

[22] X. Niu, M. Denkowski, and M. Carpuat. Bi-directional neural machine translation with synthetic parallel data. *CoRR*, abs/1805.11213, 2018.

[23] S. Pontes-Filho and M. Liwicki. Bidirectional learning for robust neural networks. *CoRR*, abs/1805.08006, 2018.

[24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. pages 652–660, jul 2017.

[25] C. R. Qi, H. Su, M. NieBner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view CNNs for object classification on 3d data. In *CVPR*, pages 5648–5656, jun 2016.

[26] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017.

[27] G. Riegler, A. O. Ulusoy, and A. Geiger. OctNet: Learning deep 3d representations at high resolutions. In *CVPR*, pages 6620–6629, jul 2017.

[28] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo. From source to target and back: Symmetric bi-directional adaptive gan. In *CVPR*, June 2018.

[29] Y. Shen, F. Chen, Y. Yang, and T. Dong. Mining point cloud local structures by kernel correlation and graph pooling. In *CVPR*, pages 4548–4557, June 2018.

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556*, 09 2014.

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[32] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, pages 945–953, dec 2015.

[33] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, June 2015.

[34] L. P. Tchapmi, C. B. Choy, I. Armeni, J. Y. Gwak, and S. Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *3DV*, 2017.

[35] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas. KPConv: Flexible and deformable convolution for point clouds. *ICCV*, 2019.

[36] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan. Graph attention convolution for point cloud semantic segmentation. In *CVPR*, June 2019.

[37] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph CNN for learning on point clouds. *arXiv:1801.07829*, 2018.

[38] W. Wu, Z. Qi, and F. Li. PointConv: Deep convolutional networks on 3d point clouds. In *CVPR*, 2019.
[39] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d ShapeNets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, jun 2015.
[40] Y. Xia, D. He, T. Qin, L. Wang, N. Yu, T. Liu, and W. Ma. Dual learning for machine translation. *CoRR*, abs/1611.00179, 2016.
[41] Q. Xu, X. Sun, C.-Y. Wu, P. Wang, and U. Neumann. Grid-gcn for fast and scalable point cloud learning. In *CVPR*, 2020.
[42] J. H. Xu Wang and L. Ma. Exploiting local and global structure for point cloud semantic segmentation with contextual point representations. In *NeurIPS*, 2019.
[43] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *CVPR*, 2020.
[44] L. Yi, L. Guibas, V. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, A. Lu, Q. Huang, and A. Sheffer. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics*, 35:1–12, 11 2016.
[45] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia. PointWeb: Enhancing local neighborhood deatures for point cloud processing. In *CVPR*, June 2019.
[46] S.-K. Y. Zhiyuan Zhang, Binh-Son Hua. Shellnet: efficient point cloud convolutional neural networks using concentric shells statistics. *ICCV*, 2019.
[47] Özgün Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger. 3d U-Net: Learning dense volumetric segmentation from sparse annotation. In *MICCAI*, pages 424–432. 2016.

Wenkai Han received the B.Eng. degree in computer science from Xiamen University, Xiamen, China, in 2018. He is currently working toward the M.Sc.degree in the School of Information Science and Engineering, Xiamen University, Xiamen, China. His research interests include computer vision, machine learning, and 3D point clouds processing.

Hai Wu received the B.Eng. degree in information computing and science from Sichuan University of Science and Technology, Sichuan, China, in 2018. He is currently working toward the M.Sc.degree in the School of Informatics, Xiamen University, Xiamen, China. His research interests include computer vision, machine learning, and 3D point clouds processing.

Chenglu Wen received the Ph.D. degree in mechanical engineering from China Agricultural University, Beijing, China, in 2009. She is currently an Associate Professor with the Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Informatics, Xiamen University, Xiamen, China. Her current research interests include 3D point cloud processing, 3D vision, and intelligent robot. Dr. Wen has coauthored about 70 research papers published in refereed journals and proceedings. She is currently an Associate Editor of IEEE-GRSL and IEEE T-ITS.

Cheng Wang received the Ph.D. degree in information and communication engineering from the National University of Defense Technology, Changsha, China in 2002. He is currently Professor and Associate Dean of the School of Information Science and Engineering, and Executive Director of Fujian Key Laboratory of Sensing and Computing for Smart City, both at Xiamen University, China. His current research interests include remote sensing image processing, mobile LiDAR data analysis, and multi-sensor fusion. He is the Chair of the ISPRS Working Group I/6 on Multi-sensor Integration and Fusion (2016-2020), council member of the China Society of Image and Graphics. He has coauthored more than 150 papers.

Xin Li received his B.S. degree in computer science from the University of Science and Technology of China in 2003, and his M.S. and Ph.D. degrees in computer science from Stony Brook University (SUNY) in 2008. He is currently an associate professor with the School of Electrical Engineering and Computer Science and the Center for Computation and Technology, Louisiana State University, USA. He leads the Geometric and Visual Computing Laboratory at LSU. His research interests include geometric and visual data processing and analysis, computer graphics, and computer vision.